99P1891

PCT WORLD INTELLECTUAL PROPE International Bu

B1

INTERNATIONAL APPLICATION PUBLISHED UNDER WO 9608763A2

(51) International Patent Classification 6 : G06F 9/32

A2

(11) International Publication Number: WO 96/08763

(43) International Publication Date: 21 March 1996 (21.03.96)

(21) International Application Number: PCT/IB95/00686

(22) International Filing Date: 24 August 1995 (24.08.95)

(30) Priority Data:
08/308,048 16 September 1994 (16.09.94) US

(71) Applicant: PHILIPS ELECTRONICS N.V. [NL/NL]; Groenewoudseweg 1, NL-5621 BA Eindhoven (NL).

(71) Applicant *(for SE only)*: PHILIPS NORDEN AB [SE/SE]; Kottbygatan 5, Kista, S-164 85 Stockholm (SE).

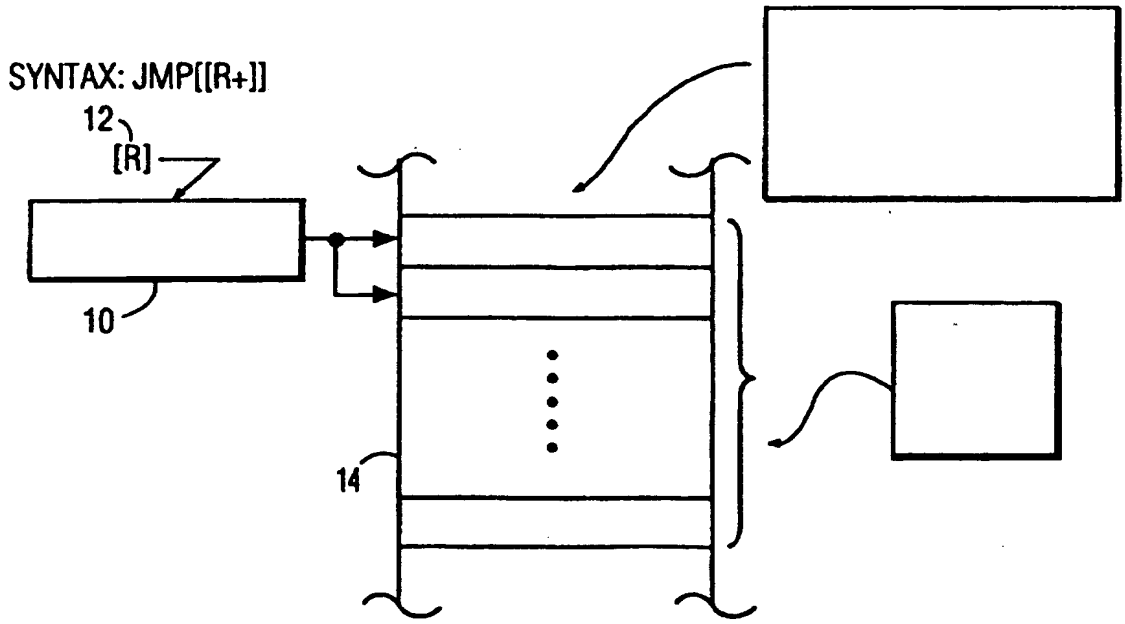
(72) Inventors: ROY, Santanu; 3529 Squirecreek Circle, San Jose, CA (US). RABELER, Thorwald; Stresemannallee 101, D-22529 Hamburg (DE).

(74) Agent: VERDONK, Peter, Lambert, Frans, Maria; Internationaal Octrooibureau B.V., P.O. Box 220, NL-5600 AE Eindhoven (NL).

(81) Designated States: JP, KR, European patent (AT, BE, CH, DE, DK, ES, FR, GB, GR, IE, IT, LU, MC, NL, PT, SE).

Published
Without international search report and to be republished upon receipt of that report.

(54) Title: METHOD, APPARATUS AND INSTRUCTION FOR PERFORMING A DOUBLE JUMP REGISTER INDIRECT OPERATION TRANSFER IN A MICROCONTROLLER



(57) Abstract

A microcontroller performs a jump d uble register indirect instruction by determining the address of a procedure to xecute using the contents of a designated pointer register as an index into a procedure address table located anywhere in the address space where each entry is an address of an entry point for the particular procedure to be executed. The address is retrieved and loaded into the microcontroller program counter resulting in a jump to the entry point of the procedure. The contents of the pointer register is post incremented to point at the next entry of the table and, as a result, the next procedure. The next execution of the instruction will cause a jump to the next procedure.

FOR THE PURPOSES OF INFORMATION ONLY

Codes used to identify States party to the PCT on the front pages of pamphlets publishing international applications under the PCT.

AT	Austria	**GB**	United Kingdom	**MR**	Mauritania
AU	Australia	**GE**	Georgia	**MW**	Malawi
BB	Barbados	**GN**	Guinea	**NE**	Niger
BE	Belgium	**GR**	Greece	**NL**	Netherlands
BF	Burkina Faso	**HU**	Hungary	**NO**	Norway
BG	Bulgaria	**IE**	Ireland	**NZ**	New Zealand
BJ	Benin	**IT**	Italy	**PL**	Poland
BR	Brazil	**JP**	Japan	**PT**	Portugal
BY	Belarus	**KE**	Kenya	**RO**	Romania
CA	Canada	**KG**	Kyrgystan	**RU**	Russian Federation
CF	Central African Republic	**KP**	Democratic People's Republic of Korea	**SD**	Sudan
CG	Congo			**SE**	Sweden
CH	Switzerland	**KR**	Republic of Korea	**SI**	Slovenia
CI	Côte d'Ivoire	**KZ**	Kazakhstan	**SK**	Slovakia
CM	Cameroon	**LI**	Liechtenstein	**SN**	Senegal
CN	China	**LK**	Sri Lanka	**TD**	Chad
CS	Czechoslovakia	**LU**	Luxembourg	**TG**	Togo
CZ	Czech Republic	**LV**	Latvia	**TJ**	Tajikistan
DE	Germany	**MC**	Monaco	**TT**	Trinidad and Tobago
DK	Denmark	**MD**	Republic of Moldova	**UA**	Ukraine
ES	Spain	**MG**	Madagascar	**US**	United States of America
FI	Finland	**ML**	Mali	**UZ**	Uzbekistan
FR	France	**MN**	Mongolia	**VN**	Viet Nam
GA	Gabon				

Method, apparatus and instruction for performing a double jump register indirect operation transfer in a microcontroller.

FIELD OF THE INVENTION

The present invention is directed to a system for performing a double jump register indirect operation and, more particularly, to a system that makes an unconditional branch to an address in instruction or code memory where the address is found in a table in memory pointed to by an address in a register designated in the instruction.

BACKGROUND ART

Today there is a demand for higher performance/cost ratio embedded microcontrollers which are required to perform ever more complex tasks. To attain this microcontrollers must execute real time code using code or instructions which occupy the minimum amount of memory. Programs typically executed by microcontrollers today are modular, that is divided into separate and some what independent subprograms/subroutines or procedures that perform different or severable tasks. These procedures are typically executed sequentially or in some known order. The procedures are also typically distributed over a wide range of memory or storage space, which can be over megabytes of address space. The procedures together perform multiple tasks. Typically the initiation of each of the procedures is performed with a call to each of the procedures. Each call is overhead intensive in that they occupy a large amount of memory space that can be expensive relative to the cost of the device in which the microcontroller is embedded.

What is needed is a technique for performing jumps to multiple procedures in a specified order that is fast and occupies a minimum of memory or storage space.

SUMMARY OF THE INVENTION

It is an object of the present invention to provide a system that calls multiple procedures or subprograms/subroutines using minimum memory space.

It is an additional object of the present invention to compress the code size of a microcontroller.

It is another object of the present invention to provide a programmer with a simple easy to use and powerful instruction for executing multiple procedures in a desired sequence.

It is also an object of the present invention to provide a microcontroller with an architecture that can perform a jump double register indirect instruction.

It is a further object of the present invention to provide a mechanism for executing procedures in a predetermined order.

The above objects can be attained by a microcontroller that performs a jump double register indirect instruction by determining the address of a procedure to execute using the contents of a designated pointer register as an index into a procedure address table located anywhere in the address space where each entry is an address of an entry point for the particular procedure to be executed. The address is retrieved and loaded into the microcontroller program counter resulting in a jump to the entry point of the procedure. The contents of the pointer register is post incremented to point at the next entry of the table and, as a result, the next procedure. The next execution of the instruction will cause a jump to the next procedure.

These, together with other objects and advantages which will be subsequently apparent, reside in the details of construction and operation as more fully hereinafter described and claimed. Reference is had to the accompanying drawings forming a part hereof, wherein like numerals refer to like parts throughout.

BRIEF DESCRIPTION OF THE DRAWINGS

Figure 1 depicts the format of the instruction of the present invention;

Figure 2 illustrates the double indirect addressing of the present invention;

Figure 3 depicts the hardware architecture of the present invention;

Figures 4 - 6 illustrate the memory arrangement of the present invention;

Figure 7 illustrates the segment select register; and

Figure 8 depicts the operations performed during execution of the instruction of the present invention.

DESCRIPTION OF THE PREFERRED EMBODIMENTS

The present invention is directed to a powerful and compact instruction designed to address the requirement for multiple procedure calls. The mnemonic for a jump double register indirect instruction according to the present invention is JMP [[Rs+]] where

"JMP" indicates the jump operation, the double brackets indicate double indirection, "Rs" designates the source register from among eight pointer registers in a register file and "+" indicates that the contents of the source register is post decremented. The two byte encoded format of this instruction 1 is illustrated in figure 1. The first byte 2 includes two nibbles where the first nibble 3 indicates operation to be performed (opcode) and the second nibble 4 indicates the address mode. The second byte 5 includes two nibbles where the first nibble 6 is an opcode extension and the second nibble 7 indicates one of the eight reqisters in the register file mentioned above. This instruction causes an unconditional branch to an address contained in a memory address or location at an address pointed to by the source register specified in the instruction where the specified register is post incremented to point to the next location in the table of procedure addresses. That is, as illustrated in figure 2, this two byte instruction can cause a unconditional jump to any address in a 24 bit linear address space (16 megabytes) indirectly pointed to by the address in one of the registers in the register file. However, it is preferred that the jump be limited to a 64k address space associated with the instruction. The address 10 in memory, which is the content of the content register 12, is used as an index into a table 14 of procedure pointers (addresses of routines). The address in the table is the jump-to address. The 24 bit address is created by combining the low order 16 bits of the program counter (PC) and either the high 8 bits of the PC or contents of the code segment (CS) register as chosen by the program through a segment select special function register. This instruction compresses code size since it can be used to index through a table of procedure addresses that are accessed in sequence. Each procedure in the use of this instruction when the next procedure is to be executed ends with this instruction allowing processing to proceed immediately with the next procedure whose address is in the table. The details of how this instruction is executed will be discussed in more detail after the details of the architecture of the preferred microcontroller are discussed in detail.

The architecture of the microcontroller system 16 of the present invention is illustrated in figure 3. This system 16 includes a single chip microcontroller 17 that performs 16 bit arithmetic operations and includes internal instruction an data storage. The microcontroller 17 supports external device 18 and 19 and, through 24 bit external address capability, supports sixteen megabytes of external instruction storage 20 and sixteen megabytes of external data storage 21. The microcontroller 17 includes a bus interface unit 22 which communicates with the external memories 20 and 21 over an external bi-directional address and data bus 24. The microcontroller 17 communicates with the external devices 18

and 19 through I/O ports 26 - 28 which are addressable as special function registers (SFR) 40. The ports 26-28, as well as other special function registers, are addressable over an internal peripheral bus 42 through the bus interface unit 22. The data memory 21 can also be accessed as off-chip memory mapped I/O through the I/O ports 26 - 28. The on-chip special function registers 40, some of which are bit addressable, also include a program status word (PSW) register 44 coupled to an interruption control unit 84 communicating with the external devices as well as the ALU the execution unit 70 and decode unit 74 for flag and general control, an interrupt register 44 timer registers 50, a system configuration register (SCR) 54 containing system configuration bits, and a special segment select (SSEL) register 56. The bus interface unit 22 isolates the peripheral special function registers 40 from the microcontroller core 60. The core 60 includes a microcode programmable execution unit 70 which controls execution of instructions by an ALU 72 and the other units. The instructions decoded by a decode unit 74 are fetched from an internal EPROM 76, which is part of the instruction memory space, or from the external instruction memory 20 by a fetch unit 78. Static RAM 80, which is part of the data memory space, as well as general purpose registers of a register file 82 are also available for instruction and data storage.

The microcontroller 17 includes a memory organization as illustrated in figures 4, 5 and 6 where figure 4 illustrates the organization into pages, figure 5 depicts the organization of a page in more detail and figure 6 illustrates the address range of the special function registers 40. As previously discussed, the microcontroller 17 has separate address spaces for instruction memory and data memory. All registers and on-chip memory are accessible (addressable) as bytes and/or words. Some dedicated data memory areas (SFR, RAM, and Register File) may also be accessed as bits (see figure 4). The term "data memory" refers to on-chip RAM 80, off-chip RAM 20 or off-chip memory mapped I/O. There are four banks of registers R0 through R7 starting at address 0 in the on-chip RAM (in the register file 82) and going up to address 1F hexadecimal. One of the four banks is selected as the active bank by two bits in the PSW. The selected bank appears as the general purpose registers. Word registers R0 - R7 are also used as address pointers during the indirect addressing, as is used in the jump double register indirect instructions, and indirect-offset addressing modes. The SFR space 40 exists in the dedicated on-chip SFR memory area in the microcontroller 17 (see figure 6). This SFR space 40 uses the upper 1K of addresses in the direct address field but is not part of the data memory map. The special function registers, such as the SSEL register 56, are in the 1K direct address block 139 from address 400 to 7FF hex. The first half 140 of this block is the on-chip SFR space. This area of SFRs

is used to access SFR mapped registers, such as the SSEL register 56, and control and data registers for on-chip peripherals and I/Os. The rest 142 is reserved for off-chip SFRs. The SFR 139 space is always directly addressed. Although the SFR 139 space uses the same addressing mode as the 1K of directed addressed data space, it is logically a separate space and should not be thought of as overlapping the indirect data space. The architecture supports RAM space segmentation into 256 pages 120 each 64K size.

Memory in the system 16 is addressed in units of bytes, each byte consisting of 8-bits. A word is a 16-bit value, consisting of two contiguous bytes. The storage order for data in the microcontroller 17 is "Little Endian", such that the lower byte of a word is stored at the lower address and the higher byte is stored at the next higher address.

As previously noted, the microcontroller 12 supports a program memory 18 with an addressable space of 16 megabytes. The instruction set includes jumps and calls, some of which operate only on the local code page, some of which can access the entire program memory space, and some of which are register indirect. Program memory target addresses referenced by jumps, calls, branches, traps and interrupts, under microcode program control, must be word aligned. However, the return address from subroutines or interrupt handlers can be on either odd or even boundaries. For instance, a branch instruction may occur at any code address, but it may only branch to an even address.

Complete programs generally consist of many different modules, segments or subroutines. However, at any given time during program execution, only a small subset of a program's segments are actually in use. Generally, this subset will include code and data. The microcontroller 17 architecture takes advantage of this by providing mechanisms to support direct access to the working set of a program's execution environment and access to additional segments on demand. At any given instant, three segments of memory are immediately accessible to an executing program. These are zero segments which include the system stack, the data segment, where the user stack and local variables reside, and the extra segment, which may be used to read remote data structures. Restricting the addressability of software modules helps gain complete control of system resources for efficient, reliable operation in a multi-tasking environment. A current working data segment 150 in the microcontroller 17 includes a 16-bit address (pointer) 152 and an 8-bit segment 154, as illustrated in figure 7. The 8-bit segment registers DS or ES hold the offset which is used to identify this current segment. These segment registers are used as extension to 16-bit pointer registers and stack pointers to allow data to be accessed through the entire 16 megabyte

address range. The ES and DS registers can be assigned consecutive addresses such that they may be addressed as a single word. There are up to sixteen 16-bit registers in the register file. Of those typically eight are accessed at one time and of the ight one is reserved for the stack pointer for the particular application and the other seven may be used as general purpose pointer registers by the application to access the different segments of the memory. A "byte" register in the SFR space contains bits that are associated with each of the seven general purpose pointer registers (i.e. not the SO)_ that selects neither DS or ES as and in the case of code the CS as the source for the most significant 8-bits for the 24-bit address. This register 156 is called the segment select register or SSEL 56 (see figure 7). Segment registers are not automatically incremented or decremented along with their associated pointer registers, but must be altered explicitly by instructions.

There are several ways in which code or instruction addresses may be formed to execute instructions on the microcontroller 17. Changing the program flow is done with simple relative branches, long relative branches, 24-bit jumps and calls, 16-bit jumps and calls, and returns. Simple relative branches use an 8-bit signed displacement added to the program counter (PC) to generate the new code address. The calculation is accomplished by shifting the 8-bit relative displacement left by one bit (since it is a displacement to a word address), sign extending the result to 24-bits, adding it to the program counter contents, and forcing the least significant bit of the result to zero. The long relative unconditional branch and call with 16-bit relative displacements use the same sequence. The branch range is +255 to -256 for 8-bit relative and +65535 to -65536 for long jump and call. Far jumps and calls include a 24-bit absolute address in the instruction and simply replace the entire program counter contents with the new value. The address range is anywhere in the 16M. Double indirect jumps use a pointer register and memory contents to determine the target address.

The jump double register indirect instruction of the present invention, as illustrated in figure 8 once it is fetched and decoded by the decode unit 74 and is presented to the execution unit 70, starts by the execution unit 70 loading (200) the contents of the source register designated in the instruction into the code address register of the fetch unit 78, so that the contents of the register can be used to fetch the procedure address. The content of the source register is also loaded into the ALU 72, incremented and stored back in the source register resulting in incrementing (202) the contents of the source register. The address loaded into the fetch unit 78 is then modified as necessary by the segment register, etc. and used to fetch (204) the procedure address from memory. This address is then loaded (206) into program counter (PC) of the fetch unit 78 and the execution of the jump double

register indirect instruction is finished. Because the PC now points at the address of the entry point of the procedure to be executed, at the beginning of the next instruction cycle the fetch unit 78 fetches the instruction of the procedure at the address loaded in the fetch unit 78 and performs normal fetch, decode and execute operations thereafter.

The many features and advantages of the invention are apparent from the detailed specification and, thus, it is intended by the appended claims to cover all such features and advantages of the invention which fall within the true spirit and scope of the invention. Further, since numerous modifications and changes will readily occur to those skilled in the art, it is not desired to limit the invention to the exact construction and operation illustrated and described, and accordingly all suitable modifications and equivalents may be resorted to, falling within the scope of the invention.

CLAIMS:

1. A method of performing an execution sequence change in a program controlled microcontroller, comprising:

- fetching a procedure address of a procedure as designated by contents of a pointer register; and

- executing the procedure designated by the procedure address.

2. A method as recited in claim 1, wherein the contents of the pointer register are incremented prior to step b.

3. A method as recited in claim 2, wherein the procedure address is located in a procedure address table and the contents of the pointer register comprises an index into the procedure address table.

4. An apparatus, comprising:

- a computer comprising:

- means for fetching a procedure address designated by contents of a computer register; and

- means for transfer of execution control to the procedure responsive to the procedure address.

5. An apparatus as recited in claim 4, further comprising a procedure address table including procedure addresses in an order of procedure execution, the contents of the register being an index into the table.

6. An apparatus as recited in claim 4, further comprising means for incrementing the contents of the computer register prior to transfer.

7 An instruction, comprising:

an operation code field designating a jump to a location designated by a procedure address in a procedure address table pointed to by a pointer register; and a register designation field associated with the operation code field and identifying the pointer register.

8. An apparatus, comprising:

- a memory including an instruction, comprising:

- an operation code field designating a jump to a location designated by a procedure address in a procedure address table pointed to by a pointer register; and

- a register designation field associated with the operation code field and identifying the pointer register;

- a computer coupled to said memory and executing a jump responsive to said instruction.

9. A method of executing computer program controlled procedures, comprising:

- providing a procedure address table including addresses of procedures to be executed;

- using contents of a register as a pointer index into the procedure address table to obtain the procedure addresses; and

- initiating the procedures of the addresses in an order of the addresses in the procedure table.

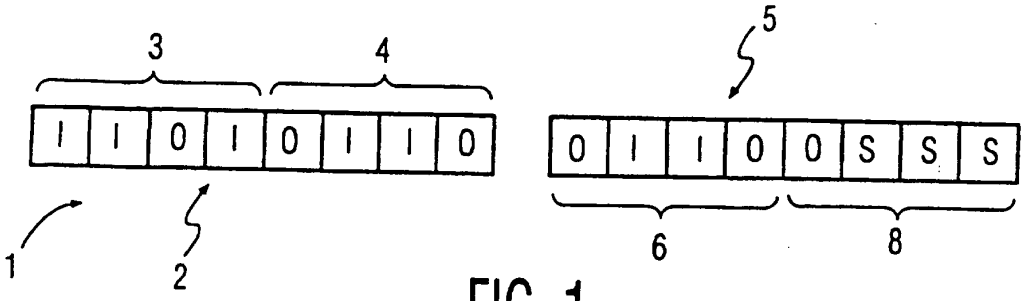


FIG. 1

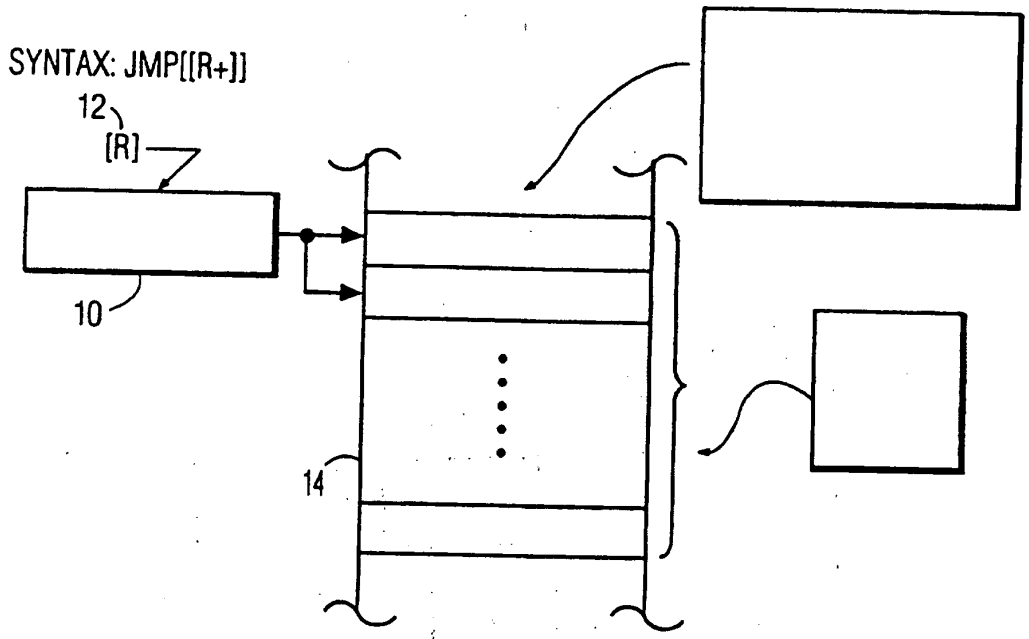


FIG. 2

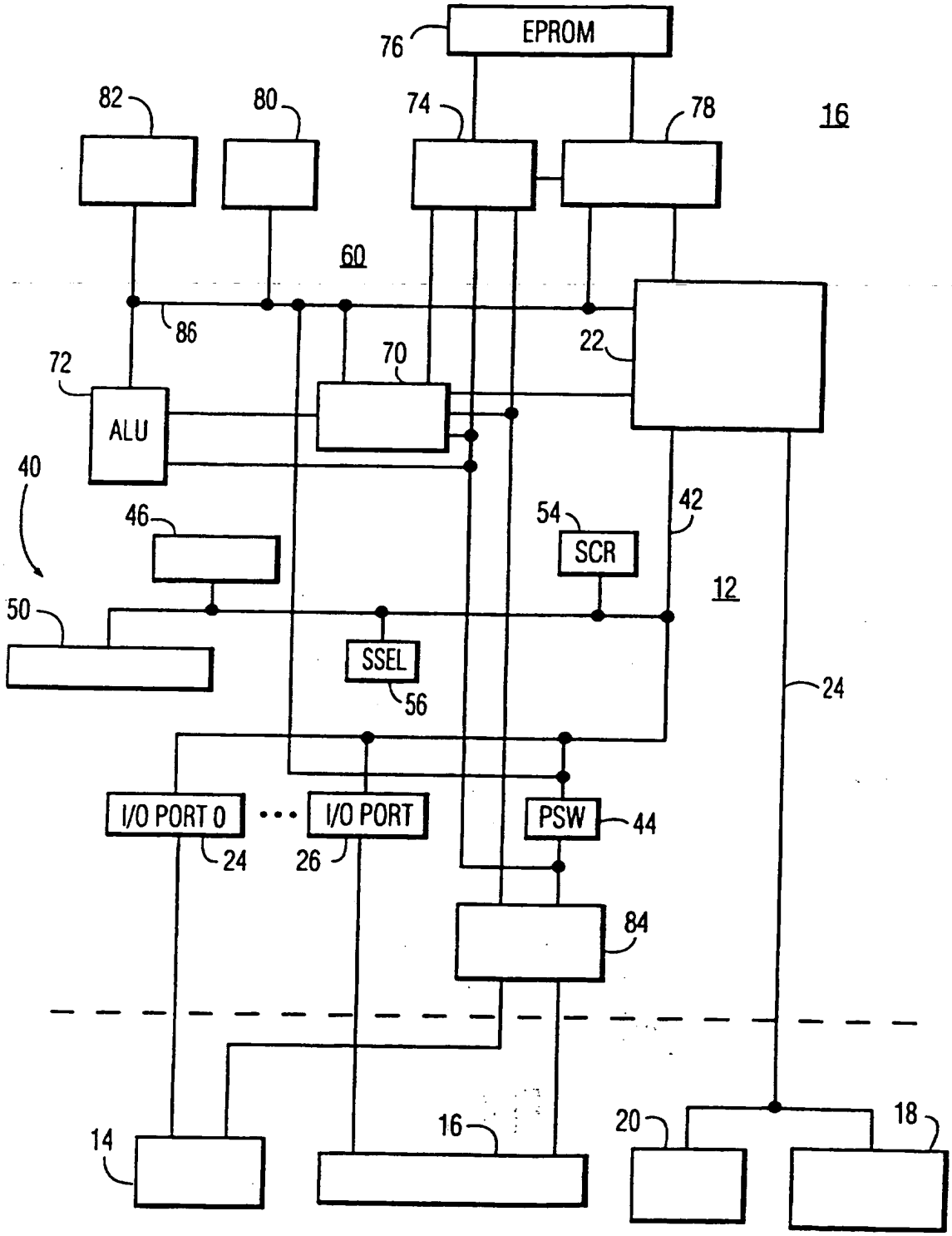


FIG. 3

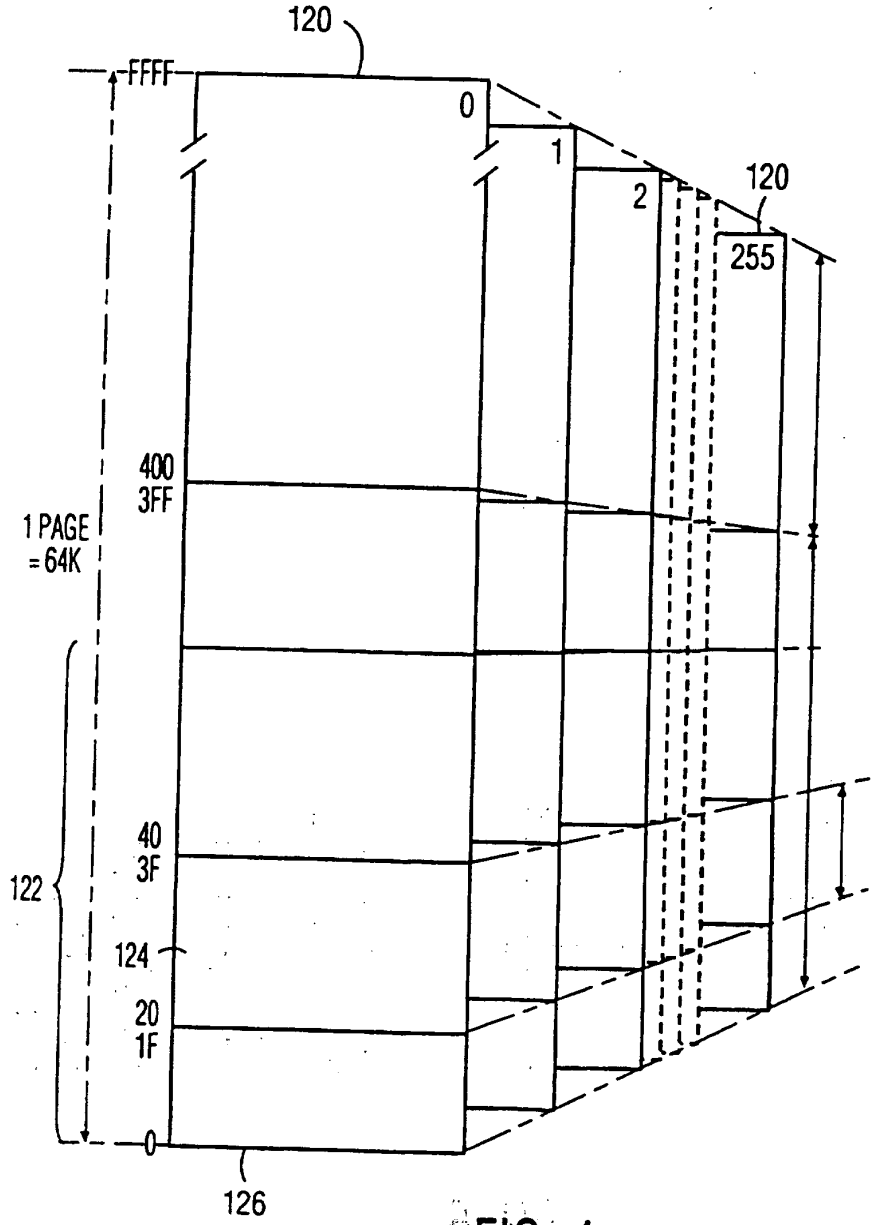


FIG. 4

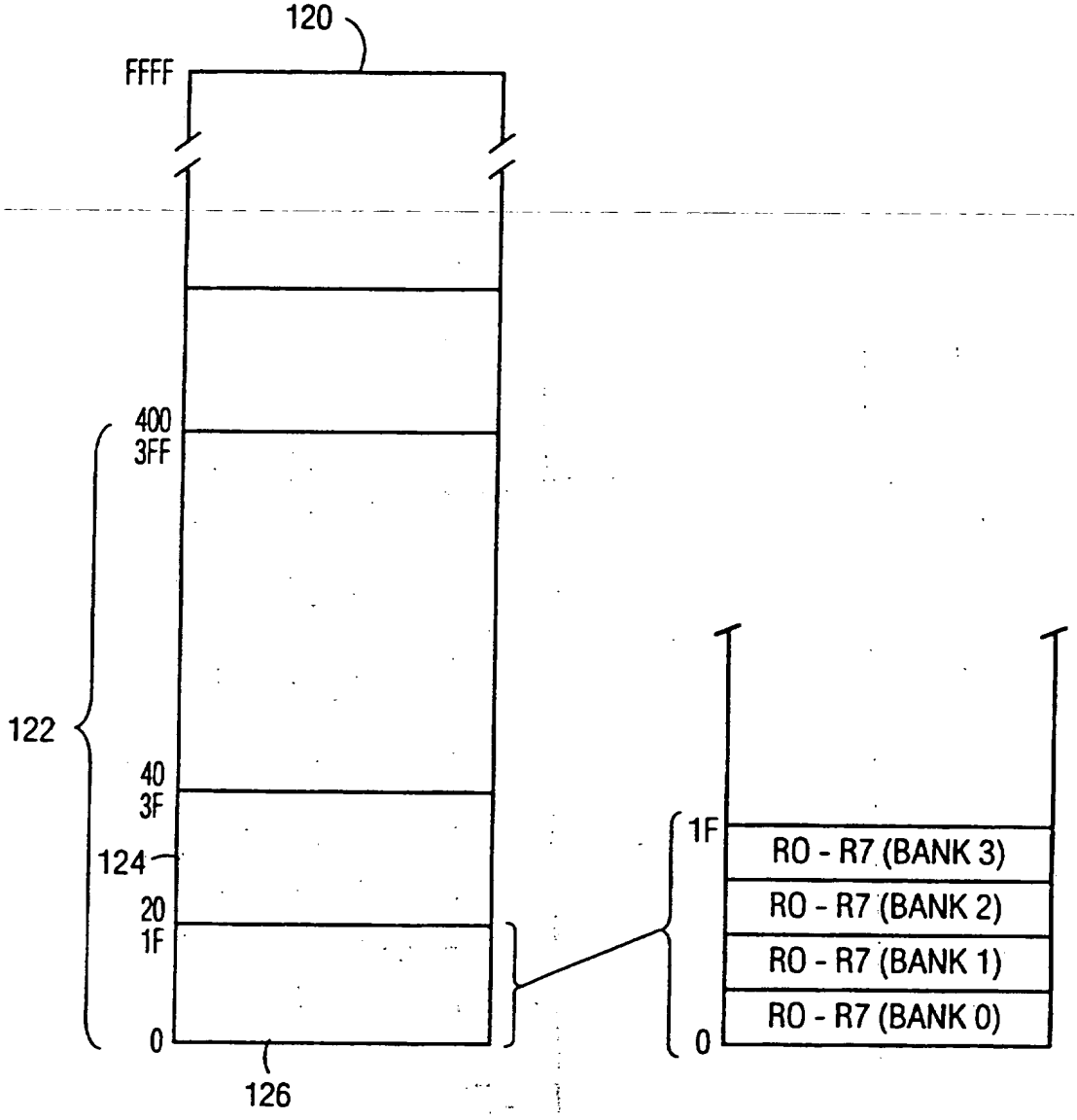


FIG. 5

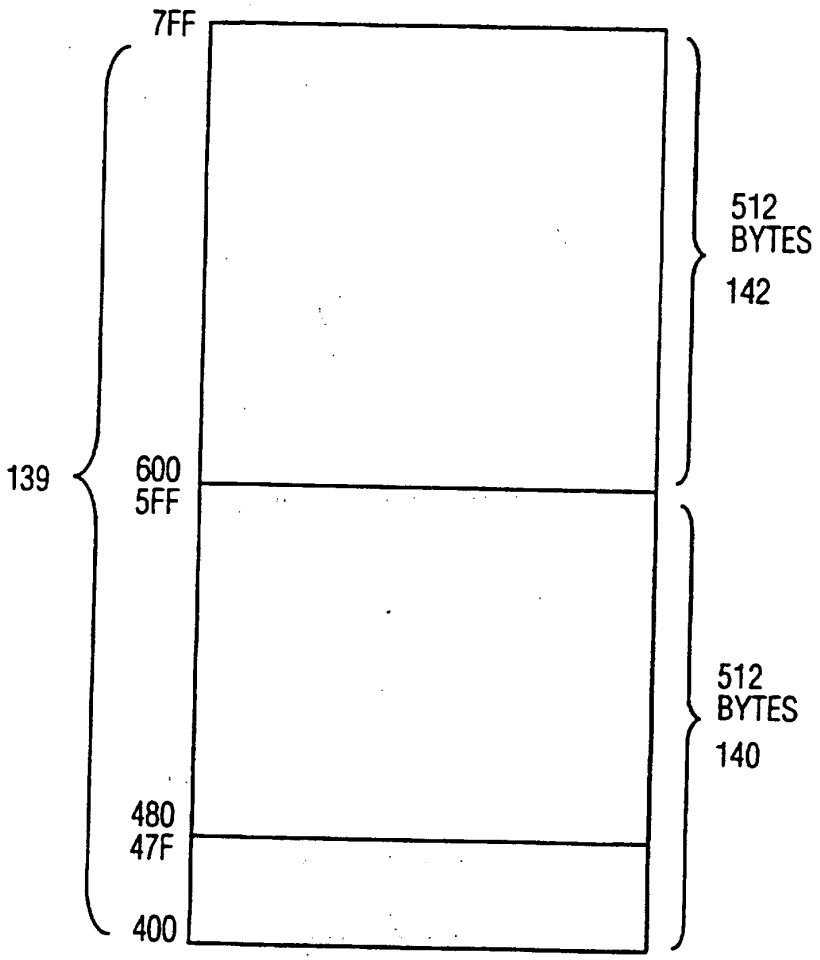


FIG. 6

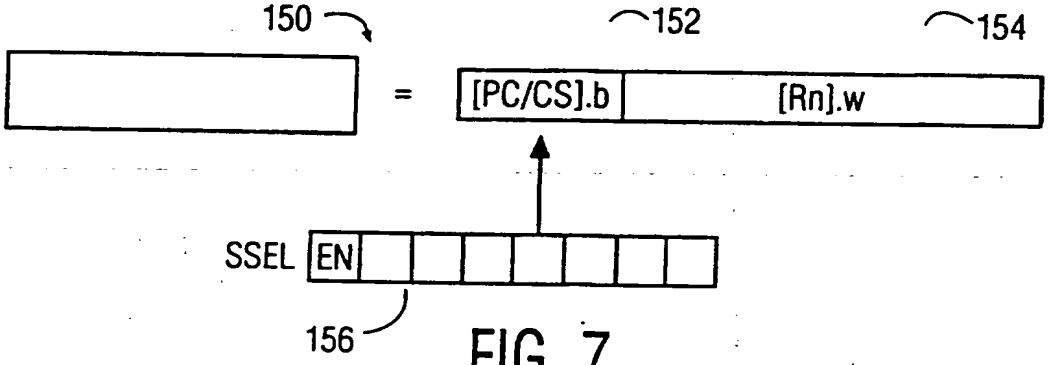


FIG. 7

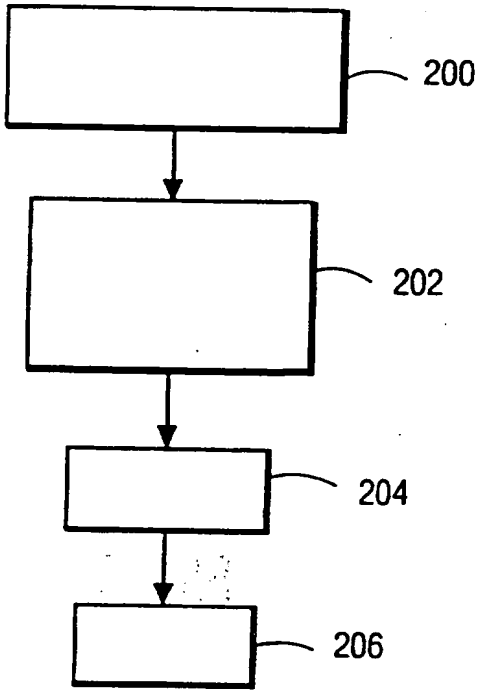


FIG. 8

PCT WORLD INTELLECTUAL PROPERTY ORGANIZATION
International Bureau

INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(51) International Patent Classification 6 : G06F 9/32	A3	(11) International Publication Number: WO 96/08763
		(43) International Publication Date: 21 March 1996 (21.03.96)

(21) International Application Number: PCT/IB95/00686

(22) International Filing Date: 24 August 1995 (24.08.95)

(30) Priority Data:
08/308,048 16 September 1994 (16.09.94) US

(71) Applicant: PHILIPS ELECTRONICS N.V. [NL/NL]; Groenewoudseweg 1, NL-5621 BA Eindhoven (NL).

(71) Applicant *(for SE only)*: PHILIPS NORDEN AB [SE/SE]; Kottbygatan 5, Kista, S-164 85 Stockholm (SE).

(72) Inventors: ROY, Santanu; 3529 Squirecreek Circle, San Jose, CA (US). RABELER, Thorwald; Stresemannallee 101, D-22529 Hamburg (DE).

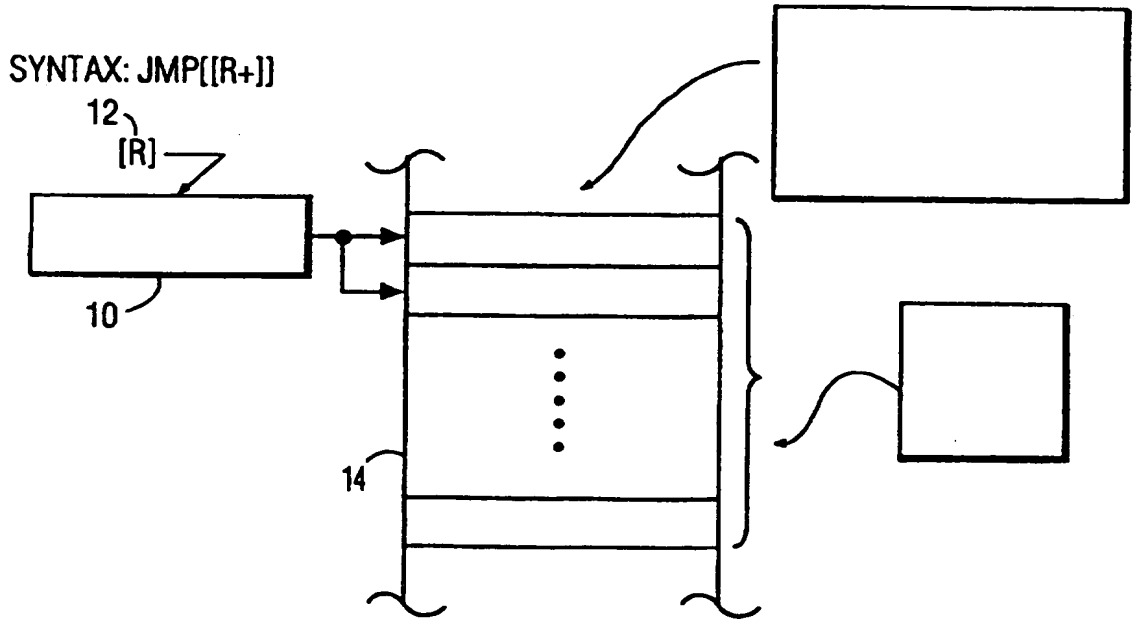
(74) Agent: VERDONK, Peter, Lambert, Frans, Maria; Internationaal Octrooibureau B.V., P.O. Box 220, NL-5600 AE Eindhoven (NL).

(81) Designated States: JP, KR, European patent (AT, BE, CH, DE, DK, ES, FR, GB, GR, IE, IT, LU, MC, NL, PT, SE).

Published
With international search report.

(88) Date of publication of the international search report:
30 May 1996 (30.05.96)

(54) Title: METHOD, APPARATUS AND INSTRUCTION FOR PERFORMING A DOUBLE JUMP REGISTER INDIRECT OPERATION TRANSFER IN A MICROCONTROLLER



(57) Abstract

A microcontroller performs a jump double register indirect instruction by determining the address f a procedure to execute using the contents of a designated pointer register as an index into a procedure address table located anywhere in the address space where each entry is an address of an entry point for the particular procedure to be executed. The address is retrieved and loaded into the microcontroller program counter resulting in a jump to the entry point f the procedure. The contents of the pointer register is post incremented to point at the next entry f the table and, as a result, the next procedure. The next executi n of the instruction will cause a jump to the next procedure.

FOR THE PURPOSES OF INFORMATION ONLY

Codes used to identify States party to the PCT on the front pages of pamphlets publishing international applications under the PCT.

AT	Austria	**GB**	United Kingdom	**MR**	Mauritania
AU	Australia	**GE**	Georgia	**MW**	Malawi
BB	Barbados	**GN**	Guinea	**NE**	Niger
BE	Belgium	**GR**	Greece	**NL**	Netherlands
BF	Burkina Faso	**HU**	Hungary	**NO**	Norway
BG	Bulgaria	**IE**	Ireland	**NZ**	New Zealand
BJ	Benin	**IT**	Italy	**PL**	Poland
BR	Brazil	**JP**	Japan	**PT**	Portugal
BY	Belarus	**KE**	Kenya	**RO**	Romania
CA	Canada	**KG**	Kyrgystan	**RU**	Russian Federation
CF	Central African Republic	**KP**	Democratic People's Republic of Korea	**SD**	Sudan
CG	Congo			**SE**	Sweden
CH	Switzerland	**KR**	Republic of Korea	**SI**	Slovenia
CI	Côte d'Ivoire	**KZ**	Kazakhstan	**SK**	Slovakia
CM	Cameroon	**LI**	Liechtenstein	**SN**	Senegal
CN	China	**LK**	Sri Lanka	**TD**	Chad
CS	Czechoslovakia	**LU**	Luxembourg	**TG**	Togo
CZ	Czech Republic	**LV**	Latvia	**TJ**	Tajikistan
DE	Germany	**MC**	Monaco	**TT**	Trinidad and Tobago
DK	Denmark	**MD**	Republic of Moldova	**UA**	Ukraine
ES	Spain	**MG**	Madagascar	**US**	United States of America
FI	Finland	**ML**	Mali	**UZ**	Uzbekistan
FR	France	**MN**	Mongolia	**VN**	Viet Nam
GA	Gabon				

INTERNATIONAL SEARCH REPORT

International application No.
PCT/IB 95/00686

A. CLASSIFICATION OF SUBJECT MATTER

IPC6: G06F 9/32
According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)

IPC6: G06F

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched

Electronic data base consulted during the international search (name of data base and, where practicable, search terms used)

EDOC, CLAIMS

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
X	Design with Microcontrollers 1988, John B. Peatman see page 107-114 ---	1-9
X	US 4803619 A (DAVID H. BERNSTEIN ET AL), 7 February 1989 (07.02.89), column 2, line 54 - column 3, line 22, abstract ---	1-9
A	US 5029078 A (RYO IWAI), 2 July 1991 (02.07.91), claims 1-8 ---	1-9

[X] Further documents are listed in the continuation of Box C. [X] See patent family annex.

* Special categories of cited documents:
"A" document defining the general state of the art which is not considered to be of particular relevance
"E" erlier document but published on or after the international filing date
"L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)
"O" document referring to an oral disclosure, use, exhibition or other means
"P" document published prior to the international filing date but later than the priority date claimed
"T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention
"X" document of particular relevance: the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone
"Y" document of particular relevance: the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art
"&" document member of the same patent family

Date f the actual completion of the international search	Date of mailing of th international search report
12 March 1996	12 -03- 1996
Name and mailing address of the ISA/ Swedish Patent Office Box 5055, S-102 42 STOCKHOLM Facsimile No. +46 8 666 02 86	Authorized officer Jan Kossmann T lephone No. +46 8 782 25 00

Form PCT/ISA/210 (second sheet) (July 1992)

INTERNATIONAL SEARCH REPORT

International application No.
PCT/IB 95/00686

C (Continuation). DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
A	GB 2007891 A (DIGITAL EQUIPMENT CORPORATION), 23 May 1979 (23.05.79), page 1, line 91 - line 109, claim 1	1-9

INTERNATIONAL SEARCH REPORT

Information on patent family members

05/02/96

International application No.
PCT/IB 95/00686

Patent document cited in search report	Publication date	Patent family member(s)	Publication date
US-A- 4803619	07/02/89	NONE	
US-A- 5029078	02/07/91	DE-A- 3821088	05/01/89
		GB-A- 2206224	29/12/88
		JP-A- 63317824	26/12/88
GB-A- 2007891	23/05/79	AU-B,B- 517948	03/09/81
		AU-A- 4103878	01/05/80
		CA-A- 1112368	10/11/81
		DE-A,C,C 2846521	26/04/79
		FR-A,B- 2407521	25/05/79
		JP-C- 1256827	29/03/85
		JP-A- 54084945	06/07/79
		JP-B- 59031735	03/08/84